# Towards Shockingly Easy Structured Classification: A Search-based Probabilistic Online Learning Framework

**Xu Sun** xusun@pku.edu.cn
∗*MOE Key Laboratory of Computational Linguistics, Peking University*
†*School of Electronics Engineering and Computer Science, Peking University*

**Editor:**



## Abstract

There are two major approaches for structured classification. One is the probabilistic gradient-based methods such as conditional random fields (CRF), which has high accuracy but with drawbacks: slow training, and no support of search-based optimization (which is important in many cases). The other one is the search-based learning methods such as perceptrons and margin infused relaxed algorithm (MIRA), which have fast training but also with drawbacks: low accuracy, no probabilistic information, and non-convergence in real-world tasks. We propose a novel and "shockingly easy" solution, a search-based probabilistic online learning method, to address most of those issues. This method searches the output candidates, derives probabilities, and conduct efficient online learning. We show that this method is with fast training, support search-based optimization, very easy to implement, with top accuracy, with probabilities, and with theoretical guarantees of convergence. Experiments on well-known tasks show that our method has better accuracy than CRF and almost as fast training speed as perceptron and MIRA. Results also show that SAPO can easily beat the state-of-the-art systems on those highly-competitive tasks, achieving record-breaking accuracies.

**Keywords:** structured prediction, graphical model, search-based learning, online learning, convergence

## 1. Introduction

Structured classification (structured prediction) models are popularly used to solve structure dependent problems in a wide variety of application domains, including natural language processing, bioinformatics, speech recognition, and computer vision. To solve those problems, many structured classification methods have been developed, most of which are from two major categories. One is the probabilistic gradient-based learning methods such as conditional random fields (CRF) (Lafferty et al., 2001). The other category of structured classification methods are the search-based learning methods, such as margin infused relaxed algorithm (MIRA) (Crammer and Singer, 2003) and structured perceptrons (Collins, 2002). Other related work on structured classification also includes maximum margin Markov networks (Taskar et al., 2003) and structured support vector machines (Tsochantaridis et al., 2004).

As for the probabilistic gradient-based learning methods such as CRF, they have high accuracy because of the exact calculation of the gradient and probabilistic information. Nevertheless, those methods have critical drawbacks:

- First, the probabilistic gradient-based methods typically do not support search-based optimization (search-based learning), which is important in structured classification problems with complex structures. In the tasks with complex structures, the gradient computation is usually quite complicated and even intractable. This is mainly because dynamic programming for calculating gradient is hard to scale to complex structures. On the other hand, the search technique is easier to scale to complex structures. That is why the gradient-based methods like CRF are usually only applied to relatively simple structures like sequential tagging, and it is rarely used for more complex tasks with or beyond tree structures. Take the syntactic parsing task with tree structures for example, instead of CRF, most of the existing systems are based on perceptrons or MIRA, because they support search-based learning. This is because search-based learning is much simpler than gradient-based learning — just search the promising output candidates and compare them with the oracle labels and do the weight update accordingly.

- The second issue is that the training of probabilistic gradient-based methods like CRF is computationally expensive and quite slow in practice. The reason is that training the CRF model requires the gradient computation for gradient-based optimization (e.g., in stochastic gradient descent training or traditional batch training methods) (Sha and Pereira, 2003; Vishwanathan et al., 2006; Sun, 2014). The gradient computation is computationally costly, especially when the tag set is with relatively high dimension.

The other category of structured classification methods are the search-based learning methods, such as structured perceptrons and MIRA. A major advantage of those methods is that they support search-based learning, such that the gradient is not needed and the learning is done by simply searching and comparing the promising output candidates with the oracle labels, and then update the model weights accordingly. As a by-product of the avoidance of gradient computation, those methods have fast training speed compared with probabilistic gradient-based learning methods like CRF. However, there are also severe drawbacks of the existing search-based learning methods:

- First, the existing search-based learning methods like perceptrons and MIRA have relatively low accuracy, compared with the probabilistic gradient-based learning methods like CRF.

- Second, in most of the real-world tasks, those search-based learning methods are non-convergent, i.e., diverges in the training. As large margin classification models, theoretically those search-based learning methods have some convergent properties based on strict separability conditions. However, those strict separability conditions are not satisfiable in most real-world tasks, as demonstrated in many prior work (Sun et al., 2014). We will also shown in the experiments that those search-based learning methods diverges dramatically as the training goes on, such that the model accuracy goes worse and worse as training goes on.

- The existing search-based methods do not support probabilistic information. The magnitude of model weights grows dramatically as training goes on, and there is no reliable probabilistic information can be derived. We will also shown the curves of the model weight magnitude in the experiments.

To address those issues, we propose a novel and "shockingly simple" solution, a **s**earch-b**a**sed **p**robabilistic **o**nline learning framework (SAPO), which can fix almost all of those drawbacks. The proposed method searches the top-$n$ output candidates, derives probabilities based on the searched candidates, and conduct fast online learning by updating the model weights.

We show that the proposed method is of fast training speed which is comparable with perceptrons and MIRA, supports search-based optimization and no need to calculate gradient, very easy to implement, with top accuracy which is even better than CRF, with reliable probability information, and with theoretical guarantees of convergence towards the optimum given reasonable conditions. Although in current stage our experiments are more focused on linear-chain tasks, the method and the theoretical results may apply to structured classification with more complex structures, for example tree and graph structures. Experiments on well-known tasks show that our method has better accuracy than CRF and almost as fast training speed as perceptron and MIRA. Results also show that SAPO can easily beat the state-of-the-art systems on those highly-competitive tasks, achieving record-breaking accuracies.

The contributions of this work are two-fold:[1]

- On the methodology side, we propose a general purpose search-based probabilistic online learning framework SAPO for structured classification. We show that SAPO can address a variety of issues of existing methods, and with theoretical justifications. Compared with probabilistic gradient-based learning methods like CRF, the proposed method supports search-based learning such that can avoid complex gradient calculation, and with extra advantages on accuracy and training speed. Compared with search-based learning methods like perceptron and MIRA, SAPO has much higher accuracy, and with theoretical and empirical justifications of convergence — perceptron and MIRA diverge in real-world tasks, as to be shown in experiments.

- On the application side, for several important natural language processing and signal processing tasks, including part-of-speech tagging, biomedical entity recognition, phrase chunking, and activity recognition, our simple search-based learning method can easily beat the state-of-the-art systems on those highly-competitive tasks, achieving record-breaking accuracies yet with fast speed.

## 2. Proposed Method

We first describe the proposed search-based probabilistic online learning algorithm SAPO, then we compare SAPO with existing methods.

**Algorithm 1** Search-based Probabilistic Online Learning Algorithm (SAPO)

**input**: top-$n$ search parameter $n$, regularization strength $\lambda$, learning rate $\gamma$
**repeat**
  Draw a sample $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}^*)$ at random from training set $S$
  Based on $\boldsymbol{w}$, search the top-$n$ outputs $Y_n = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_n\}$
  For every $\boldsymbol{y}_k \in Y_n$, compute the probability $P_k = P(\boldsymbol{y}_k|\boldsymbol{x}, \boldsymbol{w})$
  For every $\boldsymbol{y}_k \in Y_n$, update the weights by $\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma P_k \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k)$
  For $\boldsymbol{y}^*$, update the weights by $\boldsymbol{w} \leftarrow \boldsymbol{w} + \gamma \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*)$
  Regularize the weights by $\boldsymbol{w} \leftarrow \boldsymbol{w} - \frac{\gamma\lambda}{|S|}\nabla R(\boldsymbol{w})$
**until** Convergence
**return** the learned weights $\boldsymbol{w}^*$

### 2.1 Search-based Probabilistic Online Learning

The proposed search-based probabilistic online learning algorithm SAPO has the key schemes as follows: top-$n$ search (can either be exact search or approximate search), a scheme for calculating probabilities, perceptron-style update for weights, and a regularizer on weights. We introduce the technical details of the key schemes as follows, and after that we summarize the SAPO algorithm in a Figure.

First, SAPO draws a training sample $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}^*)$ at random from training set $S$, and search for the top-$n$ outputs:

$$Y_n = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_n\}$$

There are many methods to realize top-$n$ search. One method uses the A* search algorithm (Hart et al., 1968). An $A^*$ search algorithm with a Viterbi heuristic function can be used to produce top-$n$ outputs one-by-one in a efficient manner. We use the backward Viterbi algorithm (Viterbi, 1967) to compute the admissible heuristic function for the forward-style $A^*$ search. In this way we can produce the top-$n$ taggings efficiently.[2]

Then, for every $\boldsymbol{y}_k \in Y_n$, compute the probability with a log-linear fashion:

$$P_k \triangleq P(\boldsymbol{y}_k|\boldsymbol{x}, \boldsymbol{w}) \triangleq \frac{\exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k)]}{\sum_{\forall \boldsymbol{y} \in Y_n} \exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y})]} \tag{1}$$

where $\boldsymbol{w}$ is the vector of the model weights, $\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k)$ is the feature vector based on $\boldsymbol{x}$ and $\boldsymbol{y}_k$, and $Y_n$ is simply the top-$n$ outputs defined before. With this definition, we can see that $\sum_{k=1}^{n} P_k = 1$. That is, we use top-$n$ search results to estimate the probability distribution, which is typically defined as (e.g., in CRF):

$$P(\boldsymbol{y}_k|\boldsymbol{x}, \boldsymbol{w}) \triangleq \frac{\exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k)]}{\sum_{\forall \boldsymbol{y}} \exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y})]} \tag{2}$$

1. The SAPO code will be released at `http://klcl.pku.edu.cn/member/sunxu/code.htm`
2. Note that, although our search is "exact" top-$n$ search, actually "exact" top-$n$ search is not strictly required in the SAPO framework. In other words, we can replace exact A* search with non-exact beam search scheme for the SAPO algorithm. In experiments we tested both exact A* search and non-exact beam search with pruning (beam size is 50), and we find that there is almost no difference on the experimental results.

As we can see, the only difference is the normalizer — we use top-$n$ search results to estimate the normalizer. With the growth of $n$, this probability estimation in (1) goes to more and more accurate towards the traditional probability in (2). On the theoretical side, we will show in theoretical analysis that this probability estimation can be arbitrary-close to the traditional probability by using a proper $n$, and the SAPO algorithm is guaranteed to converge towards the optimum weights $\boldsymbol{w}^*$ with an arbitrary-close distance, given reasonable conditions. On the empirical side, we will show in experiments that the probability estimation is good enough for most real-world tasks even with $n = 5$ or $n = 10$.

After that, SAPO updates the weights with a perceptron fashion. For every $\boldsymbol{y}_k \in Y_n$, the weights are updated as follows:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma P_k \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k) \tag{3}$$

As we can see, this is similar to the perceptron update, except with an additional learning rate $\gamma$ and a probabilistic scaler $P_k$. On the other hand, for the oracle tagging $\boldsymbol{y}^*$, the weights are updated by

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \gamma \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*) \tag{4}$$

As we can see, this is also similar to the perceptron update. There is no need to use a probability scaler here, because the probability is 1 here.

Finally, SAPO uses a weight regularizer with regularization strength $\lambda$, just like the stochastic regularization adopted in stochastic gradient descent (SGD) (Bottou and LeCun, 2003; Niu et al., 2011; Sun et al., 2014). Following the regularization scheme of SGD, the regularization strength turns to $\lambda/|S|$ in the online learning setting (Bottou and LeCun, 2003; Niu et al., 2011; Sun et al., 2014). Also, the regularization should be scaled with the learning rate $\gamma$. Thus, by using a regularizer denoted as $R(\boldsymbol{w})$, the regularization step is as follows:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \frac{\gamma\lambda}{|S|}\nabla R(\boldsymbol{w}) \tag{5}$$

The regularizer $R(\boldsymbol{w})$ can be $L_2$, $L_1$, or other alternative regularization terms. For simplicity, in this work we use the most widely used $L_2$ regularizer (a Gaussian prior).

To sum up, the SAPO algorithm is summarized in Figure 1.

### 2.2 Comparison and Discussion

Among the existing structured classification methods, the most similar and related methods to SAPO are structured perceptrons (Collins, 2002) and CRF (Lafferty et al., 2001).

If we compare SAPO with the structured perceptron (Collins, 2002) and CRF (Lafferty et al., 2001) with stochastic training, it is interesting to see that SAPO is like a "unification" of the perceptron and the stochastically trained CRF. If we neglect the learning rate and the regularizer term of SAPO, the perceptron algorithm (Collins, 2002) can be seen as an extreme case of SAPO with $n = 1$ (i.e., using top-1 search instead of top-$n$ search). On the other hand, the stochastically trained CRF can be seen as another extreme case of SAPO with exponentially big $n$ to enumerate all possible output taggings (the only difference is that CRF uses dynamic programming instead of top-$n$ search).

In other words, perceptron can be seen as SAPO with extremely small $n$, and CRF can be seen as SAPO with extremely big $n$. We argue that SAPO is more natural than

both perceptrons and CRF — we should use a moderate value of $n$ instead of an extremely small $n$ (perceptrons) or an extremely huge $n$ (CRF). As we will show in experiments and theoretical analysis, an extremely small $n$ like perceptron will lead to the low accuracy and non-convergent training, and an extremely huge $n$ like CRF will also lead to loss of accuracy (due to the overfitting of probabilities) and high computational cost. In practice, we find it is good enough to use $n = 5$ or $n = 10$ for real-world tasks.

The MIRA algorithm also has a variation of Nbest MIRA which also uses top-$n$ search (Crammer and Singer, 2003), and interestingly, it is also good enough to use $n = 5$ or $n = 10$ for Nbest MIRA (Crammer and Singer, 2003; McDonald et al., 2005; Chiang, 2012). Nevertheless, SAPO is substantially different compared with Nbest MIRA. The major difference is that SAPO has probability estimation of different outputs while Nbest MIRA has not. Nbest MIRA treat different outputs equally without probability difference, and this is why CRF cannot be seen as a special case of Nbest MIRA. Even if Nbest-MIRA uses extremely huge $n$ in top-$n$ search, it is not equivalent to CRF and the difference is substantial. Also, there are other differences between SAPO and Nbest MIRA. For example, SAPO has the regularizer term and the learning rate, and has no need to use the "minimum change" optimization criterion of MIRA during weight update.

## 3. Theoretical Analysis

Here we give theoretical analysis on the objective function, update term, convergence conditions, and convergence rate.

### 3.1 Objective Function and Update Term

Here we analyze the equivalent objective function of SAPO and the update term of SAPO. The SAPO algorithm (Algorithm 1) is a search-based optimization algorithm, so that there is no need to compute the gradient of an objective function, and there is no explicit objective function used in the SAPO algorithm. Nevertheless, interestingly, we show that the SAPO algorithm is convergent and it converges towards the optimum weights $\boldsymbol{w}^*$ which maximizes the objective function as follows:[3]

$$\text{maximize}_{\boldsymbol{w}} \sum_{i=1}^{m} \log P(\boldsymbol{y}_i^* | \boldsymbol{x}_i, \boldsymbol{w}) - \lambda R(\boldsymbol{w}) \tag{6}$$

where $m$ is the number of training samples, i.e., $m = |S|$, and $R(\boldsymbol{w})$ is a weight regularization term for controlling overfitting. This objective function is similar to the objective function of CRF. Equivalently, for the convenience of convex-based analysis, we denote the objective function $f(\boldsymbol{w})$ as the negative form of (6):

$$f(\boldsymbol{w}) = -\sum_{i=1}^{m} \log P(\boldsymbol{y}_i^* | \boldsymbol{x}_i, \boldsymbol{w}) + \lambda R(\boldsymbol{w}) \tag{7}$$

3. The subscript of $\boldsymbol{y}$ is overloaded here. For clarity throughout, $\boldsymbol{y}$ with subscript $i$ and usually with the $*$ mark refers to the tagging of the $i$'th indexed training sample (e.g., $\boldsymbol{y}_i^*$), and $\boldsymbol{y}$ with subscript $k$ refers to the $k$'th output of the search (e.g., $\boldsymbol{y}_k$).

We show that the SAPO algorithm converges towards the optimum $\boldsymbol{w}^*$ which minimizes the convex objective function of $f(\boldsymbol{w})$:

$$\boldsymbol{w}^* = \text{minimize}_{\boldsymbol{w}} f(\boldsymbol{w}) \tag{8}$$

To clarify the theoretical analysis, we compare SAPO with the SGD (stochastic gradient descent) training scheme. Recall that the weight update has the following form in SGD (Bottou and LeCun, 2003; Niu et al., 2011):[4]

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma \nabla f_{\boldsymbol{z}}(\boldsymbol{w}) \tag{9}$$

where $\nabla f_{\boldsymbol{z}}(\boldsymbol{w})$ is the stochastic gradient of $f(\boldsymbol{w})$ based on the sample $\boldsymbol{z}$, which has the following form if use the CRF objective function:

$$\begin{aligned} \nabla f_{\boldsymbol{z}}(\boldsymbol{w}) &= -\Big\{\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*) - \sum_{\forall \boldsymbol{y}} P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}) - \frac{\lambda}{|S|} \nabla R(\boldsymbol{w})\Big\} \\ &= -\Big\{\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*) - \sum_{\forall \boldsymbol{y}} \frac{\exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y})]}{\sum_{\forall \boldsymbol{y}'} \exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}')]} \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}) - \frac{\lambda}{|S|} \nabla R(\boldsymbol{w})\Big\} \end{aligned} \tag{10}$$

To make a comparison, we denote $\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})$ as the (negative) SAPO update term for a sample $\boldsymbol{z}$ such that

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma \boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w}) \tag{11}$$

Then, according to the procedure of SAPO algorithm, it is easy to check that $\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})$ has the following form:

$$\begin{aligned} \boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w}) &= -\Big\{\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*) - \sum_{k=1}^{n} P_k \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}_k) - \frac{\lambda}{|S|} \nabla R(\boldsymbol{w})\Big\} \\ &= -\Big\{\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}^*) - \sum_{\forall \boldsymbol{y} \in Y_n} \frac{\exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y})]}{\sum_{\forall \boldsymbol{y}' \in Y_n} \exp[\boldsymbol{w}^T \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}')]} \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{y}) - \frac{\lambda}{|S|} \nabla R(\boldsymbol{w})\Big\} \end{aligned} \tag{12}$$

As we can see from (10) and (12), by increasing $n$, the SAPO update term $\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})$ can be arbitrary-close to the stochastic gradient $\nabla f_{\boldsymbol{z}}(\boldsymbol{w})$. More formally, define

$$\delta_{\boldsymbol{z}}(\boldsymbol{w}) = \nabla f_{\boldsymbol{z}}(\boldsymbol{w}) - \boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w}) \tag{13}$$

Then, for any $\epsilon \geq 0$, there is at least a corresponding $n$ such that,

$$\delta_{\boldsymbol{z}}(\boldsymbol{w}) \leq \epsilon \tag{14}$$

In other words, when $n$ is increasing, the approximation is expected to be more and more accurate and finally $\delta_{\boldsymbol{z}}(\boldsymbol{w}) \leq \epsilon$.

4. In practice, SGD and SAPO can use decayed learning rate or fixed learning rate. Following (Niu et al., 2011; Sun, 2014), for the convenience of theoretical analysis, our theoretical analysis is more focused on SGD and SAPO with fixed learning rate.

### 3.2 Optimum, Convergence, and Convergence Rate

Recall that $f(\boldsymbol{w})$ is the structured classification objective function, and $\boldsymbol{w} \in \mathcal{W}$ is the weight vector. By considering the time stamp $t$, the SAPO update (11) can be reformulated as follows:

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \gamma \boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t) \tag{15}$$

To state our convergence analysis results, we need several assumptions following Nemirovski et al. (2009). We assume $f$ is strongly convex with modulus $c$, that is, $\forall \boldsymbol{w}, \boldsymbol{w}' \in \mathcal{W}$,

$$f(\boldsymbol{w}') \geq f(\boldsymbol{w}) + (\boldsymbol{w}' - \boldsymbol{w})^T \nabla f(\boldsymbol{w}) + \frac{c}{2}||\boldsymbol{w}' - \boldsymbol{w}||^2 \tag{16}$$

where $||\cdot||$ means 2-norm $||\cdot||_2$ by default in this work. When $f$ is strongly convex, there is a global optimum/minimizer $\boldsymbol{w}^*$. We also assume Lipschitz continuous differentiability of $\nabla f$ with the constant $q$, that is, $\forall \boldsymbol{w}, \boldsymbol{w}' \in \mathcal{W}$,

$$||\nabla f(\boldsymbol{w}') - \nabla f(\boldsymbol{w})|| \leq q||\boldsymbol{w}' - \boldsymbol{w}|| \tag{17}$$

Also, let the norm of $\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})$ is bounded by $\kappa \in \mathbb{R}^+$:

$$||\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})|| \leq \kappa \tag{18}$$

Moreover, it is reasonable to assume

$$\gamma c < 1 \tag{19}$$

because even the ordinary gradient descent methods will diverge if $\gamma c > 1$ (Niu et al., 2011).

Based on the conditions, we show that SAPO converges towards the minimum $\boldsymbol{w}^*$ of $f(\boldsymbol{w})$ with an arbitrary-close distance, and the convergence rate is given as follows.

**Theorem 1 (Optimum, convergence, and rate)** *With the conditions (16), (17), (18), (19), let $\epsilon > 0$ be a target degree of convergence. Let $\tau$ be an approximation-based bound from $\boldsymbol{s}(\boldsymbol{w})$ to $\nabla f(\boldsymbol{w})$ such that*

$$[\nabla f(\boldsymbol{w}) - \boldsymbol{s}(\boldsymbol{w})]^T (\boldsymbol{w} - \boldsymbol{w}^*) \leq \tau \tag{20}$$

*where $\boldsymbol{w}$ is a historical weight vector that updated during SAPO training, and $\boldsymbol{s}(\boldsymbol{w})$ is expected $\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})$ over $\boldsymbol{z}$ such that $\boldsymbol{s}(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{z}}[\boldsymbol{s}_{\boldsymbol{z}}(\boldsymbol{w})]$. Since $\boldsymbol{s}(\boldsymbol{w})$ can be arbitrary-close to $\nabla f(\boldsymbol{w})$ by increasing $n$, SAPO can use the smallest $n$ as far as the following holds:*

$$\tau \leq \frac{c\epsilon}{2q} \tag{21}$$

*Let $\gamma$ be a learning rate as*

$$\gamma = \frac{c\epsilon - 2\tau q}{\beta q \kappa^2} \tag{22}$$

*where we can set $\beta$ as any value as far as $\beta \geq 1$. Let $t$ be the smallest integer satisfying*

$$t \geq \frac{\beta q \kappa^2 \log(q a_0 / \epsilon)}{c(c\epsilon - 2\tau q)} \tag{23}$$

*where $a_0$ is the initial distance such that $a_0 = ||\boldsymbol{w}_0 - \boldsymbol{w}^*||^2$. Then, after $t$ updates of $\boldsymbol{w}$, SAPO converges towards the optimum such that*

$$\mathbb{E}[f(\boldsymbol{w}_t) - f(\boldsymbol{w}^*)] \leq \epsilon \tag{24}$$

The proof is in Section 6.

This theorem shows that the approximation based learning like SAPO is also convergent towards the optimum of the objective function. Thus, we can approximate the true gradient by top-$n$ search and still keep the convergence properties, without the need to calculate exact gradients such as the training of CRF.

More specifically, the theorem shows that SAPO is able to converge towards the optimum of the objective function with arbitrary-close distance $\epsilon$, as far as the SAPO update term $\boldsymbol{s}(\boldsymbol{w})$ is a "close-enough approximation" (i.e., satisfying (21)) of the true gradient $\nabla f(\boldsymbol{w})$. Since $\boldsymbol{s}(\boldsymbol{w})$ can be arbitrary-close to $\nabla f(\boldsymbol{w})$ by increasing $n$, SAPO can use the smallest $n$ as far as the close-enough approximation (21) is achieved. In practice, we find setting $n$ as 5 or 10 is already empirically close-enough approximation in most of the real-world tasks. Moreover, the convergence rate is given in the theorem — SAPO is guaranteed to converge with $t$ updates, and $t$ is the smallest integer satisfying (23).

This analysis also explains why the perceptron algorithm (Freund and Schapire, 1999; Collins, 2002) does not converge in most of the practical tasks. As we discussed before, the perceptron algorithm can be essentially treated as extreme case of SAPO, which use the extremely small $n$ as 1. In most cases, the use of $n = 1$ does not satisfy the close-enough approximation condition of (21). Thus in most cases the perceptron algorithm has a bad approximation over the true gradient and it diverges (as we will show in experiments).

## 4. Related Work

Many structured classification methods have been developed, including probabilistic gradient-based learning methods and search-based learning methods. The probabilistic gradient-based learning methods include conditional random fields (Lafferty et al., 2001), and a variety of extensions such as dynamic conditional random fields (Sutton et al., 2004), hidden conditional random fields (Quattoni et al., 2007), and latent-dynamic conditional random fields (Morency et al., 2007).

The search-based learning methods include margin infused relaxed algorithm (Crammer and Singer, 2003), structured perceptrons (Collins, 2002), and a variety of related work in this direction such as latent structured perceptrons (Sun et al., 2009, 2013b), confidence weighted linear classification (CW) (Dredze et al., 2008), max-violation perceptrons (Yu et al., 2013). Most of the search-based learning methods are large-margin online learning methods. Other related work on structured classification also includes maximum margin Markov networks (Taskar et al., 2003) and structured support vector machines (Tsochantaridis et al., 2004).

For training structured classification models, especially probabilistic gradient-based learning methods like CRF and its variation models, the arguably most popular training method is stochastic gradient descent (SGD) (Bottou and LeCun, 2003; Zinkevich et al., 2010; Niu et al., 2011; Sun et al., 2012, 2014), which typically has faster convergence rate compared with alternative batch training methods, such as limited-memory BFGS (L-BFGS) (Nocedal and Wright, 1999) and other quasi-Newton optimization methods. The SGD training has theoretical guarantees to converge to the optimum weights given the convex objective function (e.g., CRF) (Bottou and LeCun, 2003; Zinkevich et al., 2010; Niu et al., 2011). For the search-based learning methods such as perceptrons, MIRA, and

their variation algorithms, the training scheme is usually quite simple and self-contained in the search-based learning algorithm/model (Collins, 2002; Crammer and Singer, 2003; McDonald et al., 2005).

As for dealing with overfitting, the probabilistic gradient-based learning methods typically use explicit regularization terms such as the widely used $L_2$ regularizer. Other regularization schemes include the $L_1$ regularizer (Andrew and Gao, 2007; Tsuruoka et al., 2009), the group Lasso regularization (Yuan and Lin, 2006; Martins et al., 2011), the structure regularization (Sun, 2014), and others (Quattoni et al., 2009). For the search-based learning methods like perceptrons and MIRA, the scheme to deal with overfitting is less formal compared with a regularizer, usually by using parameter averaging or voting (Collins, 2002; McDonald et al., 2005; Daumé III, 2006; Chiang, 2012).

## 5. Experiments

We describe the real-world tasks for experiments, the experimental settings, and the experimental results as follows.

### 5.1 Tasks

We conduct experiments on natural language processing tasks and signal processing tasks with quite diversified characteristics. The natural language processing tasks include (1) part-of-speech tagging, (2) biomedical named entity recognition, and (3) phrase chunking. The signal processing task is (4) sensor-based human activity recognition. The tasks (1) to (3) use boolean features and the task (4) adopts real-valued features. From tasks (1) to (4), the averaged length of samples (i.e., the number of tags per sample) is quite different, with the length of $23.9, 26.5, 46.6, 67.9$, respectively. The dimension of tags $|\mathcal{Y}|$ is also very diversified among tasks, with $|\mathcal{Y}|$ ranging from 5 to 45.

**Part-of-Speech Tagging (POS-Tag):** Part-of-Speech (POS) tagging is an important and highly competitive task in natural language processing. We use the standard benchmark dataset in prior work (Collins, 2002), which is derived from PennTreeBank corpus and uses sections 0 to 18 of the Wall Street Journal (WSJ) for training (38,219 samples), and sections 22-24 for testing (5,462 samples). Following prior work (Tsuruoka et al., 2011), we use features based on unigrams and bigrams of neighboring words, and lexical patterns of the current word, with 393,741 raw features[5] in total. Following prior work, the evaluation metric for this task is per-word accuracy.

**Biomedical Named Entity Recognition (Bio-NER):** This task is from the *BioNLP-2004 shared task*, which is for recognizing 5 kinds of biomedical named entities (*DNA*, *RNA*, etc.) on the *MEDLINE* biomedical text corpus. There are 17,484 training samples and 3,856 test samples. Following prior work (Tsuruoka et al., 2011), we use word pattern features and POS features, with 403,192 raw features in total. The evaluation metric is balanced F-score.

**Phrase Chunking (Chunking):** In the phrase chunking task, the non-recursive cores of noun phrases called base NPs are identified. The phrase chunking data is extracted from the data of the *CoNLL-2000 shallow-parsing shared task* (Sang and Buchholz, 2000). The

5. Raw features are those observation features based only on $\boldsymbol{x}$, i.e., no combination with tag information.

training set consists of 8,936 sentences, and the test set consists of 2,012 sentences. We use the feature templates based on word n-grams and part-of-speech n-grams, with 264,818 raw features in total. Following prior studies, the evaluation metric for this task is balanced F-score.

**Sensor-based Human Activity Recognition (Act-Recog):** This is a task based on real-valued sensor signals, with the data extracted from the Bao04 activity recognition dataset (Sun et al., 2013a). This task aims to recognize human activities (walking, bicycling, etc.) by using 5 biaxial sensors to collect acceleration signals of individuals, with the sampling frequency at 76.25HZ. Following prior work in activity recognition (Sun et al., 2013a), we use acceleration features, mean features, standard deviation, energy, and correlation features, with 1,228 raw features in total. There are 16,000 training samples and 4,000 test samples. Following prior work, the evaluation metric is accuracy.

### 5.2 Experimental Settings

We compared the proposed SAPO algorithm with strong baselines in existing literature, including both probabilistic gradient-based learning methods and search-based learning methods. For the probabilistic gradient-based learning methods, we choose the arguably most popular model CRF (Lafferty et al., 2001) as the baseline. The CRF is with the widely used $L_2$ regularization and is trained with the standard SGD training algorithm.

For search-based learning methods, we choose structured perceptrons (Perc) (Collins, 2002) and MIRA (Crammer and Singer, 2003), which are arguably the most popular search-based learning methods, as the baselines. In most cases, the averaged versions of perceptrons and MIRA work empirically better than naive versions of perceptron and MIRA (Collins, 2002; McDonald et al., 2005; Daumé III, 2006; Chiang, 2012). Thus we also compare SAPO with averaged versions of perceptrons and MIRA. To differentiate the naive and averaged versions, we denote them as Perc-Naive, Perc-Avg, MIRA-Naive, MIRA-Avg, respectively. Moreover, the MIRA method has the Nbest versions (Crammer and Singer, 2003; McDonald et al., 2005), which adopts top-$n$ search and update instead of Viterbi search and update. We also choose Nbest versions of MIRA as the additional baselines. We denote the Nbest MIRA with naive training as MIRA-Nbest-Naive, and denote the one with averaged training as MIRA-Nbest-Avg.

The regularization strength $\lambda$ of CRF are tuned among values $0.1, 0.5, 1, 2, 5$, and are determined on the development data provided by the standard dataset (POS-Tag) or simply via 4-fold cross validation on the training set (Bio-NER, Chunking, and Act-Recog). With this automatic tuning for regularization strength, we set $2, 5, 1$ and 5 for POS-Tag, Bio-NER, Word-Seg, and Act-Recog tasks, respectively. To give no tuning advantage to SAPO, SAPO simply uses the same regularizer and the same learning rate as CRF use. All the tuning are based on CRF, and there is no additional tuning for SAPO.

Also, the proposed SAPO algorithm use the same top-$n$ search scheme as the Nbest MIRA use. As shown in prior work (Crammer and Singer, 2003; McDonald et al., 2005; Chiang, 2012), it is good enough to use $n = 5$ or $n = 10$ for Nbest MIRA. It is also good enough to use $n = 5$ or $n = 10$ for the proposed SAPO algorithm. Thus, we set $n = 5$ for Nbest MIRA and SAPO for fast speed. All methods use the same set of features. Experiments are performed on a computer with the Intel(R) Xeon(R) 3.0GHz CPU.

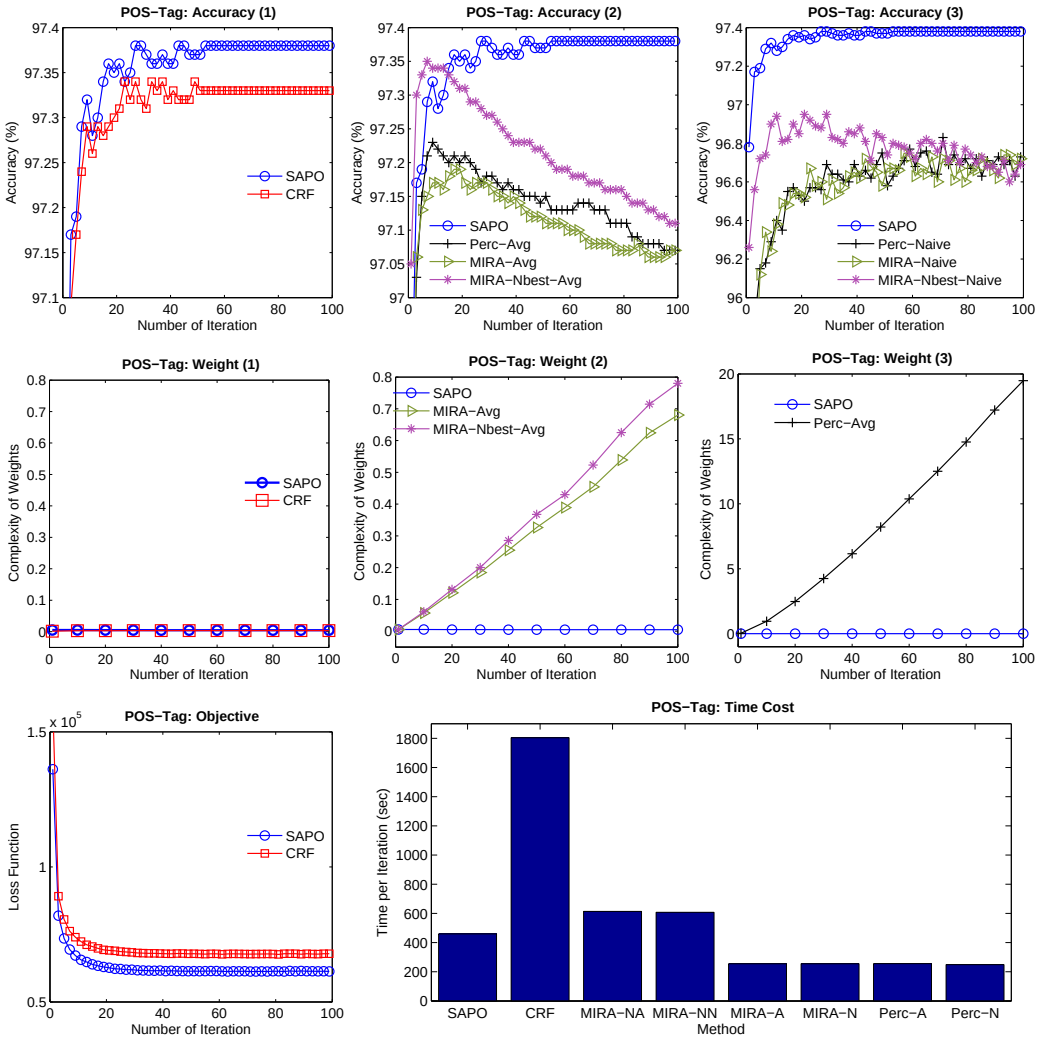


Figure 1: Results on the POS-Tag task.

### 5.3 Experimental Results

The experimental results in terms of accuracy/F-score are shown in Figure 1, Figure 2, Figure 3, and Figure 4, respectively. As we can see, although the tasks are with diversified feature types (boolean or real-value) and different characteristics, the results are quite consistent — the proposed SAPO algorithm has the best accuracies/F-scores in all of the four tasks compared with the existing baselines.

First, we compare SAPO with CRF. It is impressing that the proposed SAPO algorithm even has better accuracy than the CRF — CRF is arguably one of the most accurate models for structured classification. Note that the CRF model and other baselines are already fully optimized, which can be confirmed by comparing those results with the state-of-the-art reports on those four tasks (to be shown in a table later). As for the superiority, the reason is that the probability is distributed on top-$n$ outputs in SAPO, which is a "regularized" distribution than the probability distribution over all possible outputs (an exponential num-

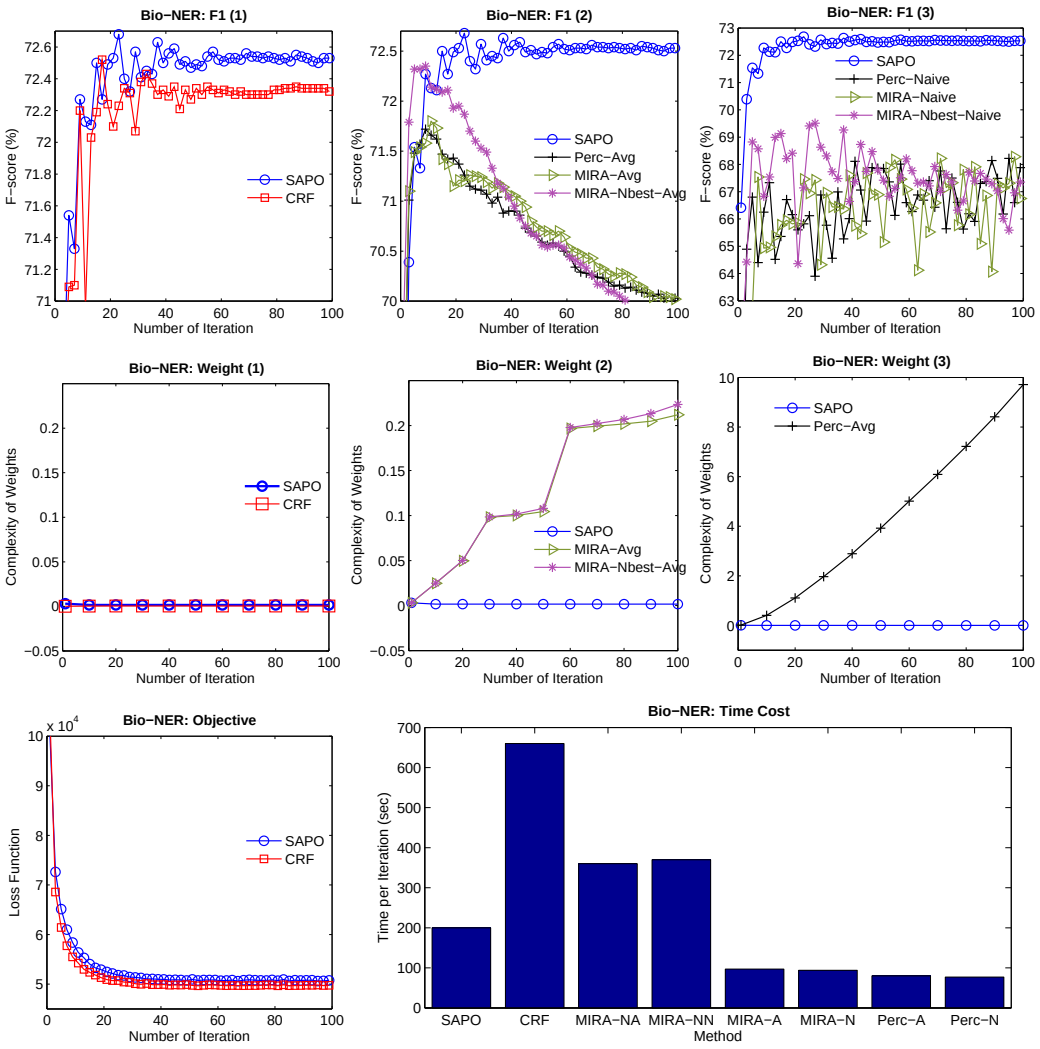


Figure 2: Results on the Bio-NER task.

ber). In this sense, SAPO is "regularizing" the exponential probability distribution to a simpler top-$n$ probability distribution. This can be seen as a probability-based regularizer with the regularization strength controlled by $n$, and interestingly the experimental results suggest that this type of regularization can indeed improve the accuracy/F-score.

We observe that SAPO is better than CRF in all of the four tasks, and we will show that in many cases the differences are statistically significant. Also, we can see that SAPO is several times faster than CRF in terms of training time. On convergence state, SAPO achieves similar or even better loss function than CRF.

Second, we compare SAPO with search-based learning methods, including naive/average versions of Perc, MIRA, and Nbest MIRA. As we can see, the superiorities of SAPO over search-based learning methods are even more significant than over CRF.

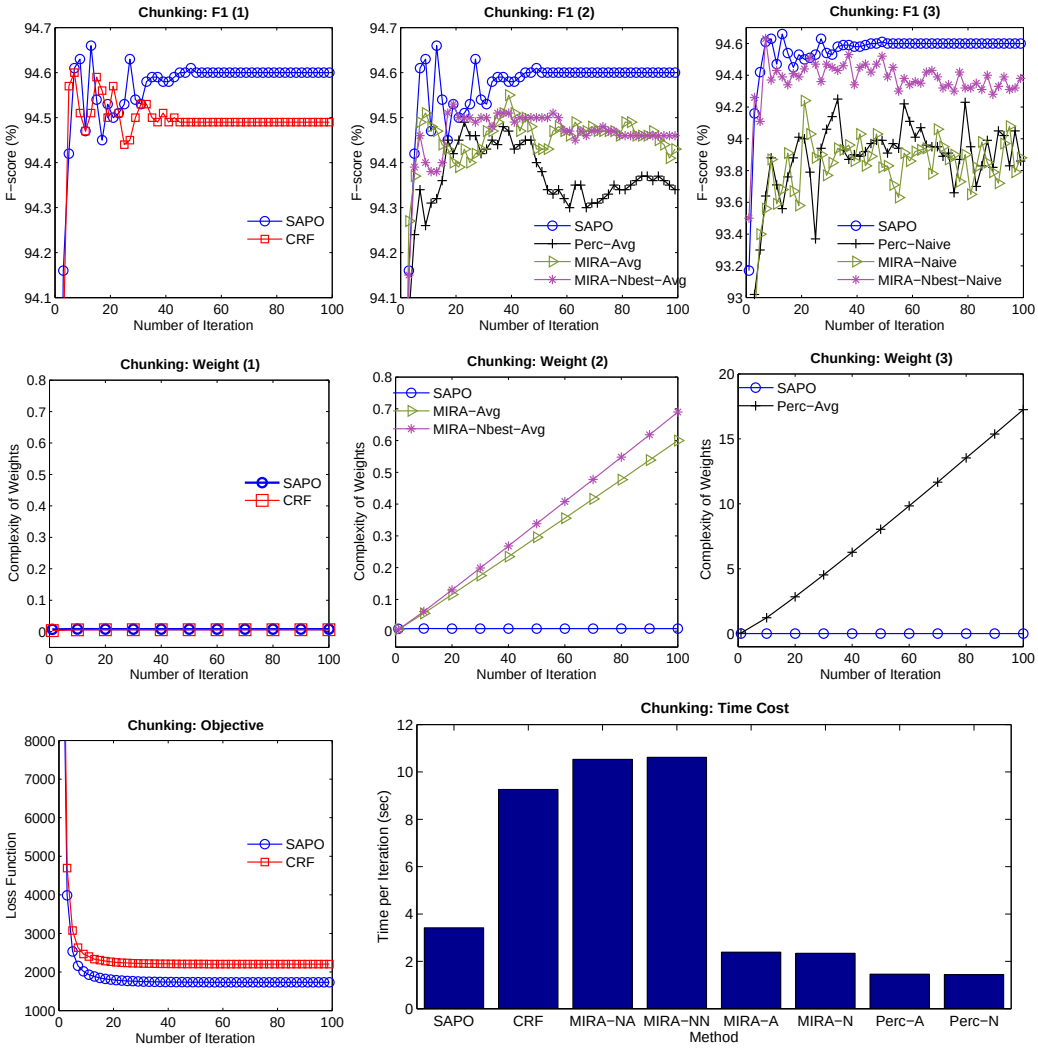


Figure 3: Results on the Chunking task.

We also conduct significance tests based on t-test.[6] For the POS-Tag task, the significance test suggests that the superiorities of SAPO over all of the baselines except CRF are very statistically significant, with at least $p < 0.01$. For the Bio-NER task, the significance test suggests that the superiorities of SAPO over all of the baselines are significant, with at least $p < 0.05$. For the Act-Recog task, the superiorities of SAPO over all of the baselines are very significant, with at least $p < 0.01$.

Our method actually outperforms the state-of-the-art records on those competitive natural language processing tasks. Those datasets are standard benchmark datasets, which can directly be compared with existing work. The POS-Tagging task is a highly competitive task, with many methods proposed, and the best report (without using extra resources) until now is achieved by using a bidirectional learning model in Shen et al. (2007), with the accuracy 97.33%. With 97.38%, our simple method achieves better accuracy compared with

6. For the tasks measured by F-score, the t-test is approximated by using accuracy to approximate F-score.

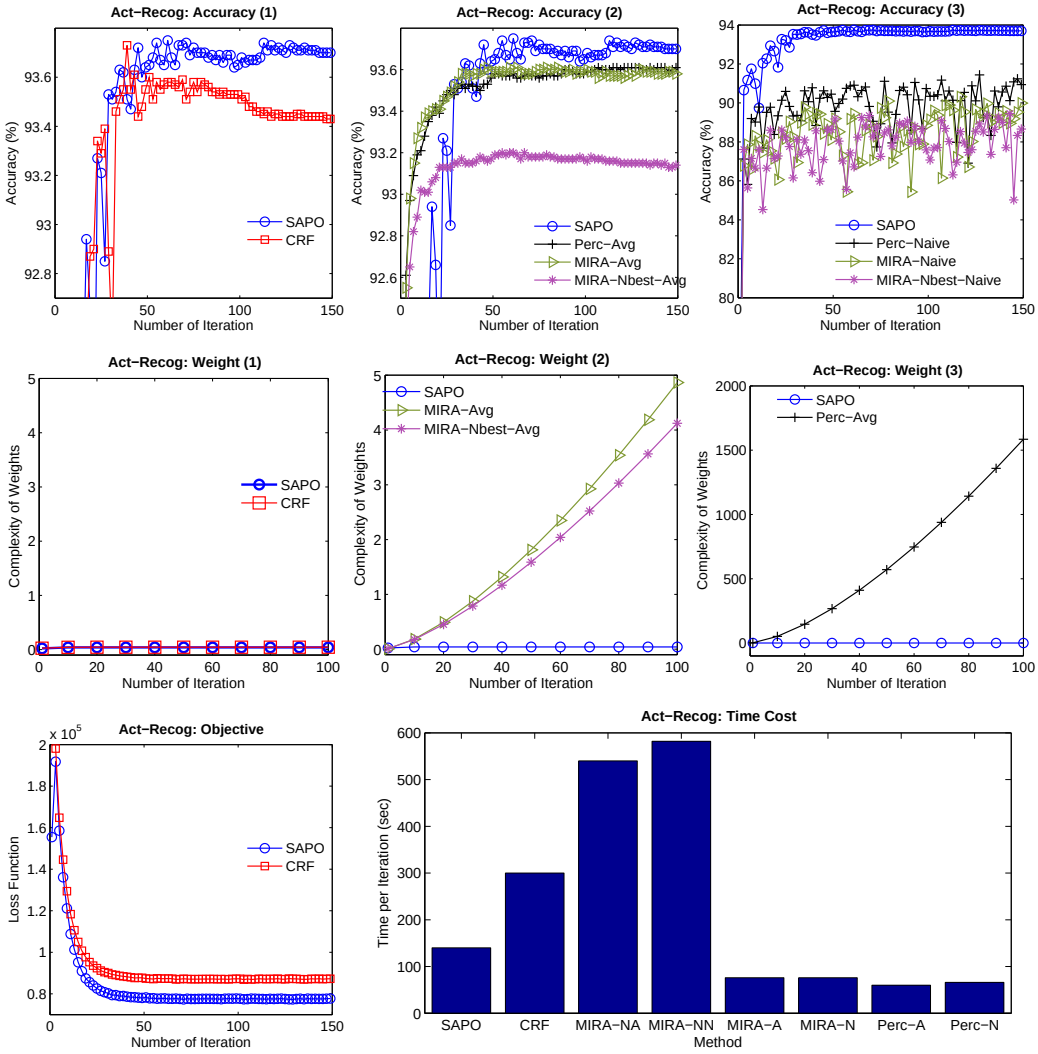


Figure 4: Results on the Act-Recog task.

all of those state-of-the-art systems. Our SAPO method also achieves or exceeds the state-of-the-art methods on the Bio-NER and Chunking tasks, which are also very competitive tasks in natural language processing communities.

The first row of those figures shows the training curves based on the number of training iterations. As we can see, SAPO and CRF is convergent as the training goes on, and Perc, MIRA, and Nbest MIRA diverge as the training goes on.

The second row of those figures shows the $\boldsymbol{w}$-complexity based on the number of training iterations. The $\boldsymbol{w}$-complexity is the averaged (absolute) value of the weights. As we can see, SAPO and CRF have convergent and very small weight complexity as the training goes on, and Perc, MIRA, and Nbest MIRA have linear or even super-linear explosion of weight complexity as the training goes on. Big weight complexity is typically a bad sign on controlling generalization risk.

The left side of the third row of those figures shows the loss function of SAPO and CRF based on the number of training iterations. As we can see, SAPO converges as good or even better in terms of the loss function, as the training goes on. This confirms our theoretical analysis on the convergence of SAPO. The right side of the third row of those figures show the training time per iteration in terms of seconds. As we can see, SAPO is with low computational cost, especially compared with CRF and Nbest MIRA.

To summarize, the experiment results demonstrate that SAPO has better accuracy than probabilistic gradient-based methods like CRF and at the same time with fast training speed like perceptrons and MIRA. Also, SAPO is convergent towards optimum and with controllable weight complexity as the training goes on. We emphasize that there are other important advantages of SAPO that are not shown in those experiments — SAPO supports search-based learning such that gradient information is not needed, gives probability information, and it is very easy to implement.

## 6. Proof

Here we give the proof of Theorem 1. First, the recursion formula is derived. Then, the bounds are derived.

### 6.1 Recursion Formula

By subtracting $\boldsymbol{w}^*$ from both sides and taking norms for (15), we have

$$\begin{aligned} ||\boldsymbol{w}_{t+1} - \boldsymbol{w}^*||^2 &= ||\boldsymbol{w}_t - \gamma \boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t) - \boldsymbol{w}^*||^2 \\ &= ||\boldsymbol{w}_t - \boldsymbol{w}^*||^2 - 2\gamma(\boldsymbol{w}_t - \boldsymbol{w}^*)^T \boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t) + \gamma^2 ||\boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t)||^2 \end{aligned} \tag{25}$$

Taking expectations and let $a_t = \mathbb{E}||\boldsymbol{w}_t - \boldsymbol{w}^*||^2$, we have

$$\begin{aligned} a_{t+1} &= a_t - 2\gamma \mathbb{E}[(\boldsymbol{w}_t - \boldsymbol{w}^*)^T \boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t)] + \gamma^2 \mathbb{E}[||\boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t)||^2] \\ &\quad \text{(based on (18) )} \\ &\leq a_t - 2\gamma \mathbb{E}[(\boldsymbol{w}_t - \boldsymbol{w}^*)^T \boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t)] + \gamma^2 \kappa^2 \\ &\quad \text{(since the random draw of } \boldsymbol{z}_t \text{ is independent of } \boldsymbol{w}_t) \\ &= a_t - 2\gamma \mathbb{E}[(\boldsymbol{w}_t - \boldsymbol{w}^*)^T \mathbb{E}_{\boldsymbol{z}_t}(\boldsymbol{s}_{\boldsymbol{z}_t}(\boldsymbol{w}_t))] + \gamma^2 \kappa^2 \\ &= a_t - 2\gamma \mathbb{E}[(\boldsymbol{w}_t - \boldsymbol{w}^*)^T \boldsymbol{s}(\boldsymbol{w}_t)] + \gamma^2 \kappa^2 \end{aligned} \tag{26}$$

We define

$$\delta(\boldsymbol{w}) = \nabla f(\boldsymbol{w}) - \boldsymbol{s}(\boldsymbol{w}) \tag{27}$$

and insert it into (16), it goes to

$$\begin{aligned} f(\boldsymbol{w}') &\geq f(\boldsymbol{w}) + (\boldsymbol{w}' - \boldsymbol{w})^T [\boldsymbol{s}(\boldsymbol{w}) + \delta(\boldsymbol{w})] + \frac{c}{2}||\boldsymbol{w}' - \boldsymbol{w}||^2 \\ &= f(\boldsymbol{w}) + (\boldsymbol{w}' - \boldsymbol{w})^T \boldsymbol{s}(\boldsymbol{w}) + \frac{c}{2}||\boldsymbol{w}' - \boldsymbol{w}||^2 + (\boldsymbol{w}' - \boldsymbol{w})^T \delta(\boldsymbol{w}) \end{aligned} \tag{28}$$

By setting $\boldsymbol{w}' = \boldsymbol{w}^*$, we further have

$$\begin{aligned} (\boldsymbol{w} - \boldsymbol{w}^*)^T \boldsymbol{s}(\boldsymbol{w}) &\geq f(\boldsymbol{w}) - f(\boldsymbol{w}^*) + \frac{c}{2}||\boldsymbol{w} - \boldsymbol{w}^*||^2 - (\boldsymbol{w} - \boldsymbol{w}^*)^T \delta(\boldsymbol{w}) \\ &\geq \frac{c}{2}||\boldsymbol{w} - \boldsymbol{w}^*||^2 - (\boldsymbol{w} - \boldsymbol{w}^*)^T \delta(\boldsymbol{w}) \end{aligned} \tag{29}$$

Combining (26) and (29), we have

$$\begin{aligned} a_{t+1} &\leq a_t - 2\gamma\mathbb{E}\Big[\frac{c}{2}||\boldsymbol{w}_t - \boldsymbol{w}^*||^2 - (\boldsymbol{w}_t - \boldsymbol{w}^*)^T\delta(\boldsymbol{w}_t)\Big] + \gamma^2\kappa^2 \\ &= (1 - c\gamma)a_t + 2\gamma\mathbb{E}[(\boldsymbol{w}_t - \boldsymbol{w}^*)^T\delta(\boldsymbol{w}_t)] + \gamma^2\kappa^2 \end{aligned} \tag{30}$$

Considering (20) and (27), it goes to

$$a_{t+1} \leq (1 - c\gamma)a_t + 2\gamma\tau + \gamma^2\kappa^2 \tag{31}$$

We can find a steady state $a_\infty$ as follows

$$a_\infty = (1 - c\gamma)a_\infty + 2\gamma\tau + \gamma^2\kappa^2 \tag{32}$$

which gives

$$a_\infty = \frac{2\tau + \gamma\kappa^2}{c} \tag{33}$$

Defining the function $A(x) = (1 - c\gamma)x + 2\gamma\tau + \gamma^2\kappa^2$, based on (31) we have

$$\begin{aligned} a_{t+1} &\leq A(a_t) \\ &\quad \text{(Taylor expansion of } A(\cdot) \text{ based on } a_\infty\text{, with } \nabla^2 A(\cdot) \text{ being 0)} \\ &= A(a_\infty) + \nabla A(a_\infty)(a_t - a_\infty) \\ &= A(a_\infty) + (1 - c\gamma)(a_t - a_\infty) \\ &= a_\infty + (1 - c\gamma)(a_t - a_\infty) \end{aligned} \tag{34}$$

Thus, we have

$$a_{t+1} - a_\infty \leq (1 - c\gamma)(a_t - a_\infty) \tag{35}$$

Unwrapping (35) goes to

$$a_t \leq (1 - c\gamma)^t(a_0 - a_\infty) + a_\infty \tag{36}$$

### 6.2 Bounds

Since $\nabla f(\boldsymbol{w})$ is Lipschitz according to (17), we have

$$f(\boldsymbol{w}) \leq f(\boldsymbol{w}') + \nabla f(\boldsymbol{w}')^T(\boldsymbol{w} - \boldsymbol{w}') + \frac{q}{2}||\boldsymbol{w} - \boldsymbol{w}'||^2$$

Setting $\boldsymbol{w}' = \boldsymbol{w}^*$, it goes to $f(\boldsymbol{w}) - f(\boldsymbol{w}^*) \leq \frac{q}{2}||\boldsymbol{w} - \boldsymbol{w}^*||^2$, such that

$$\mathbb{E}[f(\boldsymbol{w}_t) - f(\boldsymbol{w}^*)] \leq \frac{q}{2}\mathbb{E}||\boldsymbol{w}_t - \boldsymbol{w}^*||^2 = \frac{q}{2}a_t$$

In order to have

$$E[f(\boldsymbol{w}_t) - f(\boldsymbol{w}^*)] \leq \epsilon \tag{37}$$

it is required that $\frac{q}{2}a_t \leq \epsilon$, that is

$$a_t \leq \frac{2\epsilon}{q} \tag{38}$$

Combining (36) and (38), it is required that

$$(1-c\gamma)^t(a_0-a_\infty)+a_\infty \leq \frac{2\epsilon}{q} \tag{39}$$

To meet this requirement, it is sufficient to set the learning rate $\gamma$ such that both terms on the left side are less than $\frac{\epsilon}{q}$. For the requirement of the second term $a_\infty \leq \frac{\epsilon}{q}$, recalling (33), it goes to

$$\gamma \leq \frac{c\epsilon - 2\tau q}{q\kappa^2}$$

Thus, introducing a real value $\beta \geq 1$, we can set $\gamma$ as

$$\gamma = \frac{c\epsilon - 2\tau q}{\beta q\kappa^2} \tag{40}$$

Note that, to make this formula meaningful, it is required that

$$c\epsilon - 2\tau q \geq 0$$

Thus, it is required that

$$\tau \leq \frac{c\epsilon}{2q}$$

which is solved by the condition of (21).

On the other hand, we analyze the requirement of the first term that

$$(1-c\gamma)^t(a_0-a_\infty) \leq \frac{\epsilon}{q} \tag{41}$$

Since $a_0 - a_\infty \leq a_0$, it holds by requiring

$$(1-c\gamma)^t a_0 \leq \frac{\epsilon}{q} \tag{42}$$

which goes to

$$t \geq \frac{\log \frac{\epsilon}{qa_0}}{\log(1-c\gamma)} \tag{43}$$

Since $\log(1-c\gamma) \leq -c\gamma$ given (19), and that $\log \frac{\epsilon}{qa_0}$ is a negative term, we have

$$\frac{\log \frac{\epsilon}{qa_0}}{\log(1-c\gamma)} \leq \frac{\log \frac{\epsilon}{qa_0}}{-c\gamma}$$

Thus, (43) holds by requiring

$$\begin{aligned} t &\geq \frac{\log \frac{\epsilon}{qa_0}}{-c\gamma} \\ &= \frac{\log(qa_0/\epsilon)}{c\gamma} \end{aligned} \tag{44}$$

Combining (40) and (44), it goes to

$$t \geq \frac{\beta q\kappa^2 \log(qa_0/\epsilon)}{c(c\epsilon - 2\tau q)}$$

which completes the proof. ■

## 7. Conclusions and Future Work

The exiting structured classification methods are problematic. The existing probabilistic gradient-based methods such as CRF have slow training and do not support search-based optimization. The existing search-based learning methods such as perceptrons and MIRA have relatively low accuracy and is non-convergent in most of the real-world tasks. We propose a novel and "shockingly easy" solution, a search-based probabilistic online learning framework SAPO, to address all of those issues. SAPO is with fast training, support search-based optimization, very easy to implement, with top accuracy, with probability information, and with theoretical guarantees of convergence.

Although currently we more focus on sequence structures, the method and the theories can apply to structured classification with more complex structures, for example trees and graphs. Experiments on well-known benchmark tasks demonstrate that SAPO has better accuracy than CRF and roughly as fast training speed as perceptrons and MIRA. Results also show that SAPO can easily beat the state-of-the-art systems on those highly-competitive tasks, achieving record-breaking accuracies.

In current implementation, our top-$n$ search uses a simple A* search algorithm with Viterbi heuristics. This top-$n$ search algorithm is not fully optimized for speed. There are several other top-$n$ search algorithms possibly with faster speed. In the future we can optimize the top-$n$ search algorithm. We believe this can further improve the training speed of SAPO. Moreover, SAPO is a general purpose algorithm for structured classification with arbitrary structures. In the future we can apply SAPO to structured classification with more complex structures, e.g., syntactic parsing and statistical machine translation.

## Acknowledgments

This work was supported in part by National Natural Science Foundation of China (No. 61300063).

## References

Galen Andrew and Jianfeng Gao. Scalable training of $L_1$-regularized log-linear models. In *International Conference on Machine Learning (ICML)*, pages 33–40, 2007.

Léon Bottou and Yann LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2003.

David Chiang. Hope and fear for discriminative training of statistical translation models. *Journal of Machine Learning Research*, 13:1159–1187, 2012.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8, 2002.

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991, 2003.

Hal Daumé III. *Practical Structured Learning Techniques for Natural Language Processing*. PhD thesis, University of Southern California, 2006.

Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *International Conference on Machine Learning (ICML)*, pages 264–271, 2008.

Yoav Freund and Robert Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost path. *IEEE Trans. On System Science and Cybernetics*, SSC-4(2):100–107, 1968.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

Andr F. T. Martins, Noah A. Smith, Mrio A. T. Figueiredo, and Pedro M. Q. Aguiar. Structured sparsity in structured prediction. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1500–1511, 2011.

Ryan T. McDonald, Koby Crammer, and Fernando C. N. Pereira. Online large-margin training of dependency parsers. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2005.

Louis-Philippe Morency, Ariadna Quattoni, and Trevor Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *Proceedings of CVPR'07*, pages 1–8, 2007.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Feng Niu, Benjamin Recht, Christopher Re, and Stephen J. Wright. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*, pages 693–701, 2011.

Jorge Nocedal and Stephen J. Wright. Numerical optimization. *Springer*, 1999.

Ariadna Quattoni, Sybor Wang, Louis-Philippe Morency, Michael Collins, and Trevor Darrell. Hidden conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1848–1852, 2007.

Ariadna Quattoni, Xavier Carreras, Michael Collins, and Trevor Darrell. An efficient projection for l1,infinity regularization. In *International Conference on Machine Learning (ICML)*, page 108, 2009.

Erik Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL'00*, pages 127–132, 2000.

Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 134–141, 2003.

Libin Shen, Giorgio Satta, and Aravind K. Joshi. Guided learning for bidirectional sequence classification. In *Proceedings of ACL'07*, 2007.

Xu Sun. Structure regularization for structured prediction. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2402–2410. 2014.

Xu Sun, Takuya Matsuzaki, Daisuke Okanohara, and Jun'ichi Tsujii. Latent variable perceptron algorithm for structured classification. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1236–1242, 2009.

Xu Sun, Houfeng Wang, and Wenjie Li. Fast online training with frequency-adaptive learning rates for chinese word segmentation and new word detection. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 253–262, 2012.

Xu Sun, Hisashi Kashima, and Naonori Ueda. Large-scale personalized human activity recognition using online multitask learning. *IEEE Trans. Knowl. Data Eng.*, 25(11): 2551–2563, 2013a.

Xu Sun, Takuya Matsuzaki, and Wenjie Li. Latent structured perceptrons for large-scale learning with hidden information. *IEEE Trans. Knowl. Data Eng.*, 25(9):2063–2075, 2013b.

Xu Sun, Wenjie Li, Houfeng Wang, and Qin Lu. Feature-frequency-adaptive on-line training for fast and accurate natural language processing. *Computational Linguistics*, 40(3):563–586, 2014.

Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *International Conference on Machine Learning (ICML)*, 2004.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning (ICML)*, pages 823–830, 2004.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 477–485, 2009.

Yoshimasa Tsuruoka, Yusuke Miyao, and Junichi Kazama. Learning with lookahead: Can history-based models rival globally optimized models? In *Conference on Computational Natural Language Learning (CoNLL)*, 2011.

S.V.N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. Accelerated training of conditional random fields with stochastic meta-descent. In *International Conference on Machine Learning (ICML)*, pages 969–976, 2006.

Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. Max-violation perceptron and forced decoding for scalable mt training. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1112–1123, 2013.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

Martin Zinkevich, Markus Weimer, Alexander J. Smola, and Lihong Li. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2595–2603, 2010.